R Reference Card

by Tom Short, EPRI PACE, tshort@epri-pace.com 2004-10-21

Granted to the public domain. See www.Rpad.org for the source and latest version. Includes material from R for Beginners by Emmanuel Paradis (with permission).

Getting help
Most R functions have online documentation.

help(topic) documentation on topic
?topic id.
help.search("topic") search the help system
 apropos("topic") the names of all objects in the search list matching the regular expression "topic"
 help.start() start the HTML version of help
str(a) displays the internal structure of an R object
summary(a) gives a "summary" of a, usually a statistical summary but it is
generic meaning it has different operations for different classes of a
ls() show objects in the search path; specify pat="pat" to search on a pattern
ls.str() str() for each variable in the search path
dir() show files in the current directory
methods(a) shows S3 methods of class
methods(class=class(a)) lists all the methods to handle objects of class a

Input and output
load() load the datasets written with save
data(x) loads specified data objects
library(x) load add-on packages
read.table(file) reads a file in table format and creates a data frame from it; the default separator sep="" is any whitespace; use header=TRUE to read the first line as a header of column names; use as.is=TRUE to prevent character vectors from being converted to factors; use comment.char="" to prevent # from being interpreted as a comment; use skip=n to skip n lines before reading data; see the help for options on row naming, NA treatment, and others
read.csv("filename",header=TRUE) id. for columns
read.fwf(file,widths,header=FALSE,sep=" ") read a table of fixed width formatted data into a data.frame; widths is an integer vector giving the widths of the fixed-width fields
read.delim("filename",header=TRUE) id. with defaults set for reading comma-delimited files
read.delim("filename",header=FALSE) id. but with defaults set for reading tab-delimited files
read.fwf(file,widths,header=FALSE,sep=" ") as.is=FALSE read a table of fixed width formatted data into a data.frame
read.csv("clipboard") id. for columns

print(a,...) prints its arguments; generic, meaning it can have different methods for different objects
format(...,x=x) format an R object for pretty printing
write.table(x,as.is=TRUE,...,row.names=TRUE,col.names=TRUE,sep=" ") writes the columns of the matrix x to file file as a table with names
write.csv(x,"filename",row.names=TRUE,col.names=TRUE,...) writes the columns of the matrix x to file filename as a table with names

Slicing and extracting data
Indexing vectors
x[n] nth element
x[-n] all but the nth element
x[1:n] first n elements
x[-(1:n)] elements from n+1 to the end
x[n1:4] elements from n1 to 4
x[["name"]]) all elements greater than 3
x[x > 3] all elements between 3 and 5
x[x %in% c("a","and","the")] elements in the given set

NA
x[-c("a","and","the")] elements in the given set

Variable conversion
as.array(x), as.data.frame(x), as.numeric(x),
as.logical(x), as.complex(x), as.character(x),...
convert type; for a complete list, use methods(as)

Variable information
is.na(x), is.null(x), is.array(x), is.data.frame(x),
is.numeric(x), is.complex(x), is.character(x),...
for data types; for a complete list, use methods(is)
length(x) number of elements in x

dim(x) Retrieve or set the dimension of an object; dim(x) <- c(3,2)
dimnames(x) Retrieve or set the dimension names of an object
ncol(x) number of rows; NROW(x) is the same but treats a vector as a one row matrix
ncol(x) and NCOL(x) id. for columns
class(x) get or set the class of x; class(x) <- "myclass"
unclass(x) remove the class attribute of x
attr(x,which) get or set the attribute which of x
attributes(obj) get or set the list of attributes of obj

Data selection and manipulation
which.max(x) returns the index of the greatest element of x
which.min(x) returns the index of the smallest element of x
rev(x) reverses the elements of x
sort(x) sorts the elements of x in increasing order; to sort in decreasing order: rev(sort(x))
cut(x,breaks) divides x into intervals (factors); breaks is the number of cut intervals or a vector of cut points
match(x,y) returns a vector of the same length than x with the elements of y which are in y (NA otherwise)
which(x == a) returns a vector of the indices of x if the comparison operation is true (TRUE, in this example the values of 1 for which x[1] == a (the argument of this function must be a variable of mode logical)
choose(n, k) computes the combinations of k events among n repetitions
= n!/(n-k)! k!
na.omit(x) suppresses the observations with missing data (NA) (supersedes the corresponding line if x is a matrix or a data frame)
na.fail(x) returns an error message if x contains at least one NA

Data creation
c(...) generic function to combine arguments with the default forming a vector; with recursive=TRUE descends through lists combining all elements into one vector
from:to generates a sequence; "" has operator priority; 1:4 is "2,3,4,5" seq(from,to) generates a sequence by specifying increment; length= specifies desired length
seq(along=x) generates 1, 2, ..., length(x); useful for for loops
rep(x,times) replicate x times; use each= to repeat "each" element of x each times; rep(c(1,2,3),each=2) is 1 2 3 2 3
data.frame(...) create a data frame of the named or unnamed arguments; data.frame(x=1:4,ch=c("a","B","c","d"),n=10); shorter names are recycled to the length of the longest list(...) create a list of the named or unnamed arguments; list(a=c(1,2),b="hi",c=31);
array(x,dim) array with data x; specify dimensions like dim=c(3,4,2); elements of recycle if x is not long enough
matrix(x,nrow=ncol) matrix; elements of x recycle factor(x,levels) encodes a vector x as a factor
gl(n,k,length=n,labels=1:n) generate levels (factors) by specifying the pattern of their levels; k is the number of levels, and n is the number of replications
expand.grid() a data frame from all combinations of the supplied vectors or factors
rbind(...) combine arguments by rows for matrices, data frames, and others

Field separator: eol is the end-of-line separator; na is the string for missing values; use col.names=NA to add a blank column header to get the column headers aligned correctly for spreadsheet input

sink(file) output to file, until sink()
Most of the I/O functions have a file argument. This can often be a character string naming a file or a connection. file="" means the standard input or output. Connections can include files, piped, zipped files, and R variables.
On windows, the file connection can also be used with description = "clipboard". To read a table copied from Excel, use
x <- read.delim("clipboard")

To write a table to the clipboard for Excel, use
write.table(x,"clipboard",sep="\t","col.names=NA"

For database interaction, see packages RODBC, DBI, MySQL, R PostgreSQL, and ROracle. See packages XML, Rdf5, netCDF for reading other file formats.

Indexing lists
x[n] list with elements n
x[[n]] nth element of the list
x[["name"]]) element of the list named "name"

x$name id.

Indexing matrices
x[1,] element at row 1, column j
x[,1] row i
x[i,j] column j
x[,c(1,3)] columns 1 and 3
x["name",] row named "name"

Indexing data frames (matrix indexing plus the following)
x[["name"]]) column named "name"
x$name id.

Variable conversion
as.array(x), as.data.frame(x), as.numeric(x),
as.logical(x), as.complex(x), as.character(x),...
convert type; for a complete list, use methods(as)

Variable information
is.na(x), is.null(x), is.array(x), is.data.frame(x),
is.numeric(x), is.complex(x), is.character(x),...
for data types; for a complete list, use methods(is)

length(x) number of elements in x

dim(x) Retrieve or set the dimension of an object; dim(x) <- c(3,2)
dimnames(x) Retrieve or set the dimension names of an object
ncol(x) number of rows; NROW(x) is the same but treats a vector as a one row matrix
ncol(x) and NCOL(x) id. for columns

class(x) get or set the class of x; class(x) <- "myclass"

unclass(x) remove the class attribute of x

attr(x,which) get or set the attribute which of x

attributes(obj) get or set the list of attributes of obj

Data selection and manipulation
which.max(x) returns the index of the greatest element of x
which.min(x) returns the index of the smallest element of x

rev(x) reverses the elements of x

sort(x) sorts the elements of x in increasing order; to sort in decreasing order: rev(sort(x))

cut(x,breaks) divides x into intervals (factors); breaks is the number of cut intervals or a vector of cut points

match(x,y) returns a vector of the same length than x with the elements of y which are in y (NA otherwise)

which(x == a) returns a vector of the indices of x if the comparison operation is true (TRUE, in this example the values of 1 for which x[1] == a (the argument of this function must be a variable of mode logical)

choose(n, k) computes the combinations of k events among n repetitions
= n!/(n-k)! k!

na.omit(x) suppresses the observations with missing data (NA) (supersedes the corresponding line if x is a matrix or a data frame)

na.fail(x) returns an error message if x contains at least one NA
unique(x) if x is a vector or a data frame, returns a similar object but with the duplicate elements suppressed

table(x) returns a table with the numbers of the different values of x (typically for integers or factors)

subset(x, ...) returns a selection of x with respect to criteria (..., typically comparisons: x %in% x; if x is a data frame, the option select gives the variables to be kept or dropped using a minus sign

table entries as fraction of marginal table

sample(x, size) resample randomly and without replacement size elements in the vector x, the option replace = TRUE allows to resample with replacement

prop.table(x, margin=) table entries as fraction of marginal table

Arg(x) modulus; Im(x) complex conjugate

log(x, base) computes the logarithm of

cor(x) correlation matrix of

dim(x) dimensions of x

subset(x, y) factors, plots the

cumsum(x) a vector which

var(x, y) if it is a matrix or a data frame, the variance-covariance matrix is calculated

cov(x) if it is a matrix, centers and reduces the data; to center only use

sd(x) mean of the elements of

colSums(x) id. for columns

colMeans(x) id. for columns

Advanced data processing

apply(X, INDEX, FUN) = a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply to each element of the list X

tapply(X, INDEX, FUNs) apply to each cell of a ragged array given by X with indexes INDEX

by(data, INDEX, FUN) apply to data frame data subsetted by INDEX

merge(a, b, by=) merge two data frames by common columns or row names

tabs(a, b, data=x) a contingency table from cross-classifying factors

aggregate(x, by, FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ...) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with the repeated measurements in separate records; the option collapse specifies the number of records from which each wide entry is derived

Advanced data processing

apply(X, INDEX, FUN) = a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply to each element of the list X

tapply(X, INDEX, FUNs) apply to each cell of a ragged array given by X with indexes INDEX

by(data, INDEX, FUN) apply to data frame data subsetted by INDEX

merge(a, b, by=) merge two data frames by common columns or row names

tabs(a, b, data=x) a contingency table from cross-classifying factors

aggregate(x, by, FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ...) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with the repeated measurements in separate columns of the same record and 'long' format with the repeated measurements in separate records; use (direction="wide") or (direction="long")

Advanced data processing

apply(X, INDEX, FUN) = a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply to each element of the list X

tapply(X, INDEX, FUNs) apply to each cell of a ragged array given by X with indexes INDEX

by(data, INDEX, FUN) apply to data frame data subsetted by INDEX

merge(a, b, by=) merge two data frames by common columns or row names

tabs(a, b, data=x) a contingency table from cross-classifying factors

aggregate(x, by, FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ...) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with the repeated measurements in separate columns of the same record and 'long' format with the repeated measurements in separate records; use (direction="wide") or (direction="long")

Advanced data processing

apply(X, INDEX, FUN) = a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply to each element of the list X

tapply(X, INDEX, FUNs) apply to each cell of a ragged array given by X with indexes INDEX

by(data, INDEX, FUN) apply to data frame data subsetted by INDEX

merge(a, b, by=) merge two data frames by common columns or row names

tabs(a, b, data=x) a contingency table from cross-classifying factors

aggregate(x, by, FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ...) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with the repeated measurements in separate columns of the same record and 'long' format with the repeated measurements in separate records; use (direction="wide") or (direction="long")

Advanced data processing

apply(X, INDEX, FUN) = a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply to each element of the list X

tapply(X, INDEX, FUNs) apply to each cell of a ragged array given by X with indexes INDEX

by(data, INDEX, FUN) apply to data frame data subsetted by INDEX

merge(a, b, by=) merge two data frames by common columns or row names

tabs(a, b, data=x) a contingency table from cross-classifying factors

aggregate(x, by, FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ...) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with the repeated measurements in separate columns of the same record and 'long' format with the repeated measurements in separate records; use (direction="wide") or (direction="long")

Advanced data processing

apply(X, INDEX, FUN) = a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply to each element of the list X

tapply(X, INDEX, FUNs) apply to each cell of a ragged array given by X with indexes INDEX

by(data, INDEX, FUN) apply to data frame data subsetted by INDEX

merge(a, b, by=) merge two data frames by common columns or row names

tabs(a, b, data=x) a contingency table from cross-classifying factors

aggregate(x, by, FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ...) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with the repeated measurements in separate columns of the same record and 'long' format with the repeated measurements in separate records; use (direction="wide") or (direction="long")

Advanced data processing

apply(X, INDEX, FUN) = a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply to each element of the list X

tapply(X, INDEX, FUNs) apply to each cell of a ragged array given by X with indexes INDEX

by(data, INDEX, FUN) apply to data frame data subsetted by INDEX

merge(a, b, by=) merge two data frames by common columns or row names

tabs(a, b, data=x) a contingency table from cross-classifying factors

aggregate(x, by, FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ...) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with the repeated measurements in separate columns of the same record and 'long' format with the repeated measurements in separate records; use (direction="wide") or (direction="long")

Advanced data processing

apply(X, INDEX, FUN) = a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply to each element of the list X

tapply(X, INDEX, FUNs) apply to each cell of a ragged array given by X with indexes INDEX

by(data, INDEX, FUN) apply to data frame data subsetted by INDEX

merge(a, b, by=) merge two data frames by common columns or row names

tabs(a, b, data=x) a contingency table from cross-classifying factors

aggregate(x, by, FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ...) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with the repeated measurements in separate columns of the same record and 'long' format with the repeated measurements in separate records; use (direction="wide") or (direction="long")
fourfoldplot(x) visualizes, with quarters of circles, the association between two dichotomous variables for different populations (x must be an array with dim=c(2,2), k), or a matrix with dim=c(2,2) if k = 1
assocplot(x) Cohen–Friendly graph showing the deviations from independence of rows and columns in a two dimension contingency table
mosaicplot(x) ‘mosaic’ graph of the residuals from a log-linear regression of a contingency table.
pairs(x) if x is a matrix or a data frame, draws all possible bivariate plots between the columns of x
plot.ts(x) if x is an object of class "ts", plot of x with respect to time, x may be multivariate but the series must have the same frequency and dates
ts.plot(x) id. but if x is multivariate the series may have different dates and must have the same frequency
qqnorm(x) quantiles of x with respect to the values expected under a normal law
qqplot(x, y) quantiles of y with respect to the quantiles of x
contour(x, y, z) contour plot (data are interpolated to draw the curves), x and y must be vectors and z must be a matrix so that dim(z)=c(length(x), length(y)) (x and y may be omitted)
filled.contour(x, y, z) id. but the areas between the contours are coloured, and a legend of the colours is drawn as well
image(x, y, z) draws a vertical line at abcissa x
filled.contour(x, y, z) draws a vertical line at abcissa x
contour(x, y, z) draws a horizontal line at ordinate y
image(x, y, z) draws a line of the form c(bottom, left, top, right), the default values are c(0,1, 0,1, 2,1)
mfcol a vector of the form c(nr, nc) which partitions the graphical window as a matrix of nr lines and nc columns, the plots are then drawn in columns
mrow.id. but the plots are drawn by row
pch controls the type of symbol, either an integer between 1 and 25, or any single character within "*
ps an integer which controls the size in points of texts and symbols
pty a character which specifies the type of the plotting region, "s": square, "n": maximal
tck a value which specifies the length of tick-marks on the axes as a fraction of the smallest of the width or height of the plot; if tck=1 a grid is drawn
tcl a value which specifies the length of tick-marks on the axes as a fraction of the height of a line of text (by default tcl=0.5)
xaxt if xaxt="n" the x-axis is set but not drawn (useful in conjunction with xlab=axis(1,...))
yaxt if yaxt="n" the y-axis is set but not drawn (useful in conjunction with ylab=axis(side=2,...))

Graphical parameters
These can be set globally with par(…): many can be passed as parameters to plotting commands.
adj controls text justification (0: left-justified, 0.5 centred, 1 right-justified)
bg specifies the colour of the background (ex. : bg="red", bg="blue",…)
by the list of the 657 available colours is displayed with colors()
legend(x, y, legend) adds the legend at the point (x,y) with the symbols by x and y
title() adds a title and optionally a sub-title
axis(side, vect) adds an axis at the bottom (side=1), on the left (2), at the top (3), or on the right (4); vect (optional) gives the abcissa (or ordinates) where tick-marks are drawn
cex main cex.axis, cex.label, cex.main, the title, axis, and the sub-title, cex.sub
col controls the color of symbols and lines; use color names: "red", "blue", see colors() or as #RRGGBB; see rgb(), hsv(), gray(), and rainbow(); as for cex there are: col.axis, col.lab, col.main, col.sub
font an integer which controls the style of text (1: normal, 2: italics, 3: bold, 4: bold italics); as for cex there are: font.axis, font.lab, font.main, font.sub
las an integer which controls the orientation of the axis labels (0: parallel to the x-axis, 1: horizontal, 2: perpendicular to the x-axis, 3: vertical)
lty controls the type of lines, can be an integer or string (1: "solid", 2: "dashed", 3: "dotted", 4: "dashed", 5: "longdash", 6: "twodash", or a string of up to eight characters (between "0" and "9") which specifies alternatively the length, in points or pixels, of the drawn elements and the blanks, for example lty="44" will have the same effect than lty=2

Lattice (Trellis) graphics
barchart(…x) histogram of the values of x with respect to those of x
bwplot(…x) “box-and-whiskers” plot
densityplot(…x) density functions plot
dotplot(…x) Cleveland dot plot (stacked plots line-by-line and column-by-column)
histogram(…x) histogram of the frequencies of x
qqmath(…x) quantiles of x with respect to the values expected under a theoretical distribution
stripplot(…x) single dimension plot, x must be numeric, y may be a factor
qq(…x) quantiles to compare two distributions, x must be numeric, y may be numeric, character, or factor but must have two “levels”
xyplot(…x) bivariate plots (with many functionalities)
levelplot(z~x+y) coloured plot of the values of z at the coordinates given by x and y (x, y and z are all of the same length)
spm(…x) matrix of bivariate plots
parallel(…x) parallel coordinates plot

Optimization and model fitting
Optim(par, fn, method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN") general-purpose optimization; par is initial values, fn is function to optimize (normally minimize)

lm(formula) fit linear models; formula is typically of the form response ~ terms + terms + …; use I(…y) + I(…x) for terms made of nonlinear components
glm(formula, family=) fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution; family is a description of the error distribution and link function to be used in the model; see ?family

nls(formula) nonlinear least-squares estimates of the nonlinear model parameters

approx(x, y=) linearly interpolate given data points; x can be an xy plotting structure

spline(x, y=) cubic spline interpolation

loess(formula) fit a polynomial surface using local fitting

Many of the formula-based modeling functions have several common arguments: data= the data frame for the formula variables, subset= a subset of variables used in the fit, na.action= action for missing values: "na.fail", "na.omit", or a function. The following generics often apply to model fitting functions:

predict(fit, ...) predictions from fit based on input data
df.residual(fit) returns the number of residual degrees of freedom
coef(fit) returns the estimated coefficients (sometimes with their standard-errors)
residuals(fit) returns the residuals
deviance(fit) returns the deviance
fitted(fit) returns the fitted values

logLik(fit) computes the logarithm of the likelihood and the number of parameters

AIC(fit) computes the Akaike information criterion or AIC

Statistics

aov(formula) analysis of variance model

anova(fit, ...) analysis of variance (or deviance) tables for one or more fitted model objects
density(x) kernel density estimates of x
binom.test(), pairwise.t.test(), power.t.test(), prop.test(), t.test(), ... use help.search("test")

Distributions

rnorm(n, mean=0, sd=1) Gaussian (normal)

rexp(n, rate=1) exponential

rgamma(n, shape, scale=1) gamma

rpois(n, lambda) Poisson

rweibull(n, shape, scale=1) Weibull

rcauchy(n, location=0, scale=1) Cauchy

rbeta(n, shape1, shape2) beta

rt(n, df) 'Student' (t)

rf(n, df1, df2) Fisher–Snedecor (F) (χ²)

rchisq(n, df) Pearson

rbinom(n, size, prob) binomial

rgeom(n, prob) geometric

rhyper(nn, m, n, k) hypergeometric

rlogis(n, location=0, scale=1) logistic

rlnorm(n, meanlog=0, sdlog=1) lognormal

rnbinom(n, size, prob) negative binomial

runif(n, min=0, max=1) uniform

rwilcox(nn, m, n) Wilcoxon’s statistics

All these functions can be used by replacing the letter r with d, p or q to get, respectively, the probability density (dfunc(x, ...)), the cumulative probability density (pfunc(x, ...)), and the value of quantile (qfunc(p, ...), with 0 < p < 1).

Programming

function( arglist ) expr function definition

return(value)

if(cond) expr

if(cond) cons.expr else alt.expr

for(var in seq) expr

while(cond) expr

repeat expr

break

next

Use braces {} around statements

ifelse(test, yes, no) a value with the same shape as test filled with elements from either yes or no

do.call(funnname, args) executes a function call from the name of the function and a list of arguments to be passed to it